ECOLE MAROCAINE DES SCIENCES DE L'INGENIEUR
Membre de HONORIS UNITED UNIVERSITIES

ÉCOLE MAROCAINE DES SCIENCES DE L'INGÉNIEUR

Filière Ingénierie Informatique et Réseaux

RAPPORT DE PROJET

Développement d'un module ERP sous Odoo 17 :

Gestion Informatisée des Objets Perdus

Réalisé par :
AIT SAID Mouad

Encadré par :
Pr. AITDAOUD Mohammed

Année Universitaire 2025 - 2026

Table des matières

Table des figures

Remerciements

L'aboutissement de ce projet de développement sous Odoo 17 est le fruit d'un effort constant et d'une collaboration fructueuse entre les différents acteurs du projet. Je tiens, en premier lieu, à exprimer ma profonde gratitude à mon encadrant, **Pr. AITDAOUD Mohammed**, pour la confiance qu'il m'a accordée, pour sa disponibilité exemplaire ainsi que pour ses conseils techniques d'une valeur inestimable qui ont guidé mes pas tout au long de cette réalisation. Je souhaite également adresser mes remerciements les plus sincères à l'ensemble du corps professoral de l'EMSI pour la qualité de l'enseignement dispensé et pour les outils intellectuels qu'ils nous ont fournis afin de relever les défis du monde numérique. Enfin, je ne saurais oublier ma famille pour leur soutien moral infaillible et leurs encouragements quotidiens, me permettant de travailler dans des conditions de sérénité optimales.

Résumé

Ce projet porte sur la conception, le développement et l'implémentation d'un module personnalisé au sein de l'écosystème Odoo 17, intitulé **tp_objets_perdus**. La problématique centrale repose sur l'inefficacité des méthodes traditionnelles de gestion des objets égarés, souvent basées sur des registres physiques ou des processus non centralisés. L'objectif est de proposer une solution numérique robuste permettant l'enregistrement, le suivi et la restitution des objets trouvés de manière fluide et transparente pour l'utilisateur final. L'architecture s'appuie sur un environnement conteneurisé Docker, garantissant une isolation parfaite des services et une portabilité maximale pour un déploiement professionnel. Ce travail démontre l'agilité des outils ERP modernes pour répondre à des besoins logistiques spécifiques tout en assurant l'intégrité des données grâce à la base PostgreSQL.

Abstract

This project focuses on the design, development, and implementation of a custom module within the Odoo 17 ecosystem, titled **tp_objets_perdus**. The core issue addressed is the inefficiency of traditional methods for managing lost items, often based on physical logs or non-centralized processes. The objective is to provide a robust digital solution allowing the registration, tracking, and return of found items through a streamlined and professional workflow. The architecture relies on a Docker-containerized environment, ensuring perfect service isolation and maximum portability across different server infrastructures. This work highlights the integration of Python and XML within the Odoo framework to meet specific logistical needs while ensuring data integrity and security for the institution.

Introduction Générale

Dans le paysage technologique contemporain, la transformation digitale est devenue un levier stratégique incontournable pour les organisations désireuses d'optimiser leurs processus internes. La gestion des objets égarés, bien que perçue comme une tâche administrative secondaire, représente en réalité un défi logistique et relationnel majeur pour les institutions recevant un flux constant de public. Les méthodes traditionnelles, reposant sur des registres manuscrits ou des fichiers disparates, montrent aujourd'hui leurs limites : perte d'informations critiques, manque de réactivité dans le processus de restitution et insatisfaction croissante des usagers.

C'est dans cette optique que s'inscrit le présent projet, visant à concevoir et déployer une solution numérique centralisée sous l'ERP Odoo 17. En exploitant la puissance fonctionnelle de ce framework, nous ambitionnons de transformer une gestion manuelle archaïque en un flux de travail automatisé, transparent et sécurisé. Ce projet ne se contente pas d'une simple numérisation de données ; il explore l'intégration complexe entre la logique métier en Python et les interfaces utilisateurs en XML, tout en s'appuyant sur une infrastructure moderne conteneurisée via Docker pour garantir une haute disponibilité et une scalabilité optimale.

Chapitre 1

Présentation du Projet

1.1 Contexte du projet

Le projet s'insère dans une dynamique globale de modernisation des services au sein de l'École Marocaine des Sciences de l'Ingénieur (EMSI). Avec une population estudiantine en constante croissance et une multiplication des activités sur site, la quantité d'objets oubliés ou perdus (fournitures, appareils électroniques, documents administratifs) a atteint un volume nécessitant une gestion structurée. Actuellement, le service d'accueil doit jongler entre les sollicitations des étudiants et le stockage physique des biens, sans disposer d'un outil de recherche rapide. Ce projet répond donc à un besoin réel exprimé par le personnel administratif pour fluidifier la communication entre le moment où un objet est trouvé et celui où il est remis à son propriétaire légitime, tout en professionnalisant l'image numérique de l'établissement.

1.2 Problématique

La problématique centrale réside dans l'inefficacité flagrante des systèmes de gestion non centralisés. Le manque de visibilité sur l'inventaire des objets trouvés entraîne des délais de restitution prohibitifs, voire la perte définitive de certains biens par défaut de traçabilité. Les questions fondamentales auxquelles ce projet doit répondre sont les suivantes : Comment assurer un enregistrement exhaustif et immédiat dès la découverte d'un objet ? Par quel moyen garantir l'intégrité des données pour éviter les erreurs de restitution ? Enfin, comment automatiser le cycle de vie de l'objet, depuis sa déclaration initiale jusqu'à son archivage après restitution, tout en offrant aux administrateurs un tableau de bord précis de l'activité ? L'enjeu est de passer d'un mode réactif, souvent désorganisé, à un mode proactif soutenu par une base de données relationnelle robuste.

1.3 Objectifs du projet

L'objectif primordial de ce travail est de livrer un module personnalisé, nommé `tp_objets_perdus`, parfaitement intégré à l'écosystème Odoo 17. Pour atteindre cette finalité, plusieurs objectifs spécifiques ont été définis :

- **Automatisation de la saisie :** Développer un formulaire intuitif permettant d'associer des métadonnées précises (catégorie, lieu de découverte, état de l'objet) et des preuves visuelles via l'intégration de photographies.
- **Gestion des états (Workflow) :** Implémenter une logique d'états dynamique permettant de suivre l'évolution de chaque dossier (Reçu, En attente, Restitué, Réclamé) pour une traçabilité sans faille.
- **Optimisation de la recherche :** Mettre en place des filtres avancés et des vues (Liste, Kanban) facilitant l'identification rapide d'un objet parmi des centaines d'entrées.
- **Sécurisation et Déploiement :** Garantir la persistance des données via PostgreSQL et assurer la portabilité de la solution grâce à un environnement Dockerisé, facilitant ainsi sa maintenance et son futur déploiement sur les serveurs de production.

1.4 Méthodologie

Pour garantir le succès de cette réalisation technique, nous avons adopté une méthodologie inspirée du cycle en V, adaptée aux méthodes agiles pour plus de flexibilité. La première phase a consisté en une **étude de faisabilité** et une analyse rigoureuse des besoins métier. S'en est suivie une étape de **conception logicielle** où les diagrammes UML (Cas d'utilisation, Classes, Séquence) ont été tracés pour fixer l'architecture du système. La phase de **développement** a été réalisée de manière itérative : codage des modèles Python, conception des vues XML, et configuration des droits d'accès. Enfin, une phase de **tests unitaires et fonctionnels** a permis de valider chaque fonctionnalité avant la rédaction du présent rapport. Cette approche structurée nous a permis de respecter les délais tout en assurant une qualité de code conforme aux standards de développement Odoo.

Chapitre 2

Conception

2.1 Diagramme de cas d'utilisation

Le diagramme de cas d'utilisation est une étape fondamentale pour identifier les limites du système et les interactions entre les différents acteurs. Il permet de visualiser clairement les fonctionnalités attendues telles que la création d'une fiche d'objet, la recherche multicritère et le suivi des changements d'état par l'agent. En définissant ces interactions, nous garantissons que le module répondra précisément aux besoins des utilisateurs finaux tout en sécurisant les accès.

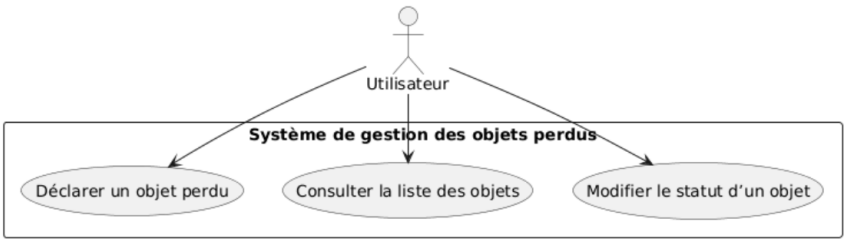

FIGURE 2.1 – Diagramme de cas d'utilisation du module

2.2 Diagramme de classes

Le diagramme de classes représente la structure statique du module en détaillant les entités et les attributs stockés en base de données PostgreSQL. Il définit comment l'ORM d'Odoo va traduire nos objets Python en tables relationnelles, incluant les champs de texte, de date et les listes de sélection. Cette modélisation est cruciale pour assurer la cohérence des données au sein de l'ERP et faciliter les futures extensions du schéma de données métier.

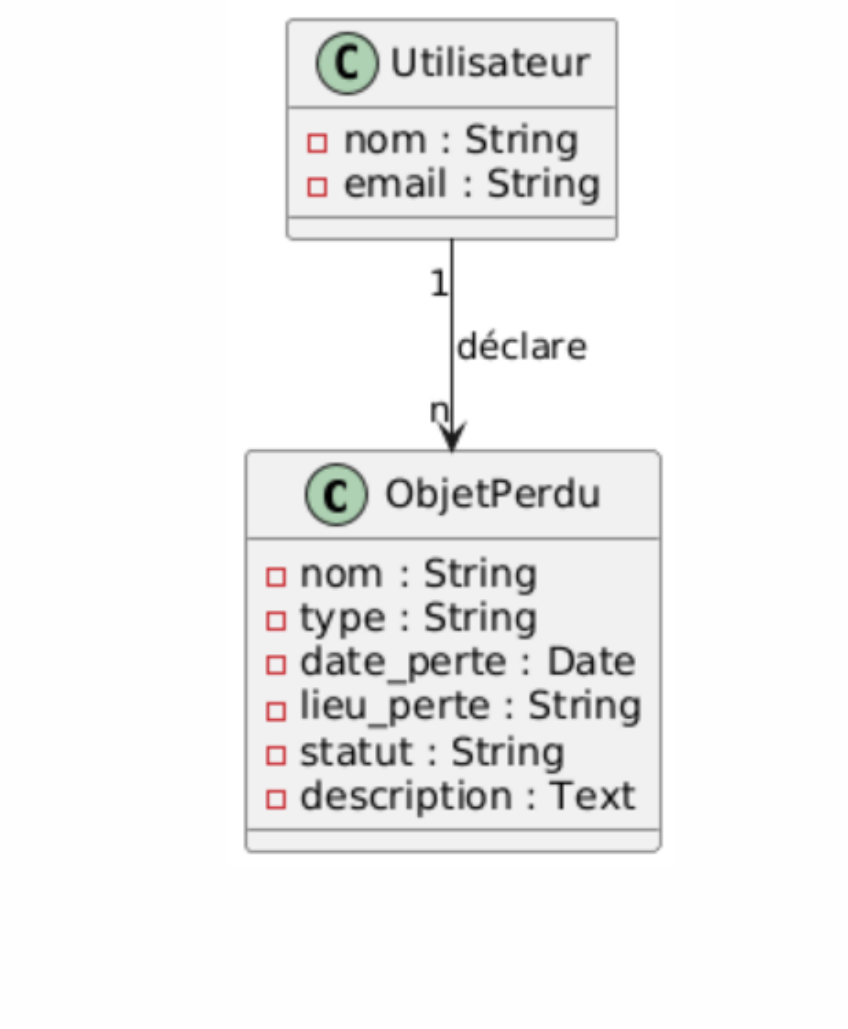

FIGURE 2.2 – Diagramme de classes du module

2.3 Diagramme de séquence

Le diagramme de séquence illustre la dynamique du système en montrant comment les messages circulent entre l'utilisateur, le serveur Odoo et la base de données. Il permet de comprendre le processus de validation interne lors de l'enregistrement d'un objet ou du passage d'un état à un autre au sein du workflow. C'est un outil indispensable pour déboguer la logique de contrôle et s'assurer que le flux d'informations respecte scrupuleusement les règles métier établies.

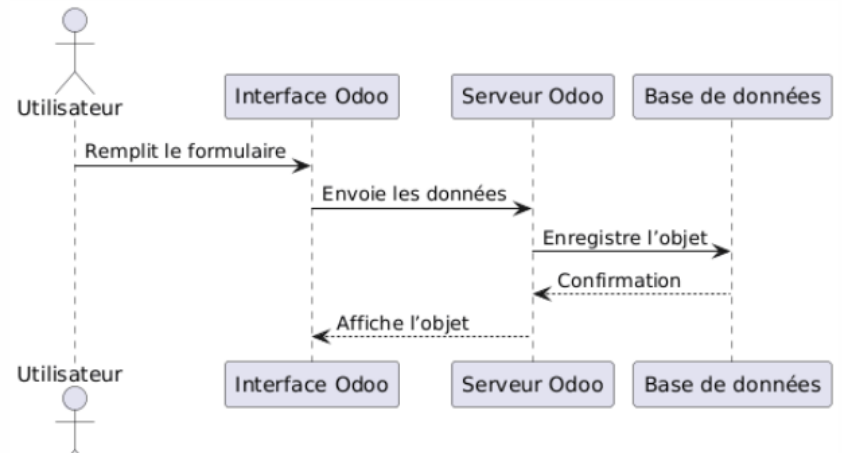

FIGURE 2.3 – Diagramme de séquence du processus de déclaration

Chapitre 3

Outils Utilisés

Pour mener à bien ce projet, nous avons mobilisé un ensemble de technologies modernes permettant de garantir la robustesse de l'application. L'architecture repose sur la virtualisation des services pour assurer une cohérence entre le développement et la production.

Odoo 17 : Framework ERP utilisé pour le développement rapide d'applications professionnelles et la gestion des vues utilisateur dynamiques.

Docker : Solution de conteneurisation permettant de déployer l'environnement de développement de manière isolée et reproductible partout.

PostgreSQL : Système de gestion de base de données relationnelle robuste utilisé par Odoo pour stocker l'intégralité des informations.

VS Code : Éditeur de code performant utilisé pour le développement des fichiers Python et XML grâce à ses extensions spécialisées.

Chapitre 4

Partie Réalisation

4.1 Structure du module dans l'environnement Docker

La structure du module est organisée selon les standards stricts d'Odoo pour permettre une reconnaissance automatique par le serveur lors du démarrage. Nous y retrouvons le dossier `models` pour la logique Python, `views` pour l'interface XML et `security` pour les fichiers CSV de droits d'accès. Cette organisation modulaire facilite la maintenance à long terme et permet d'isoler chaque aspect technique du projet pour une meilleure lisibilité globale du code source.

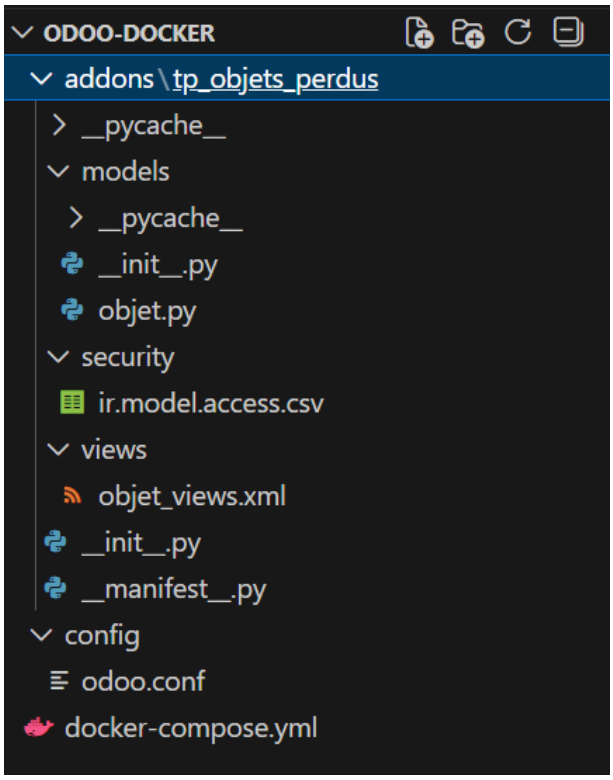

FIGURE 4.1 – Structure du module dans VS Code

4.2 Installation et activation du module

L'installation du module se fait via l'interface d'administration d'Odoo après avoir mis à jour la liste des applications dans les paramètres du système. Cette étape est cruciale car elle déclenche la lecture du fichier manifest et la création automatique des tables correspondantes dans la base de données PostgreSQL. Une fois activé, le module devient immédiatement disponible dans le menu principal d'Odoo pour les utilisateurs autorisés à gérer les objets égarés.

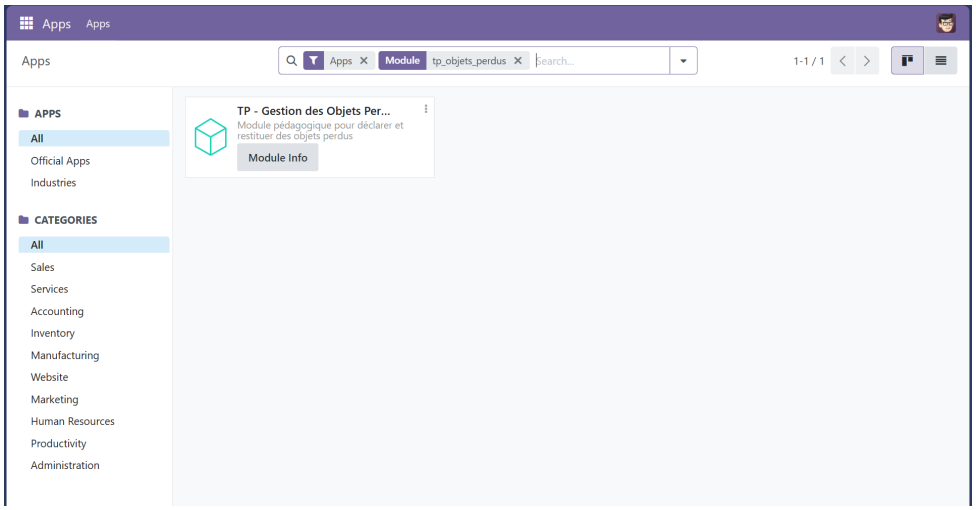

FIGURE 4.2 – Activation du module dans Odoo

4.3 Interface de déclaration d'un objet perdu

L'interface de déclaration a été conçue pour être la plus intuitive possible afin de minimiser le temps de formation des agents sur le terrain. Elle contient des champs obligatoires pour garantir que chaque objet est correctement documenté (type, lieu, date) avant d'être enregistré définitivement dans le système. Grâce aux widgets natifs d'Odoo, la sélection des catégories se fait en quelques clics, assurant une base de données parfaitement structurée et exploitable.

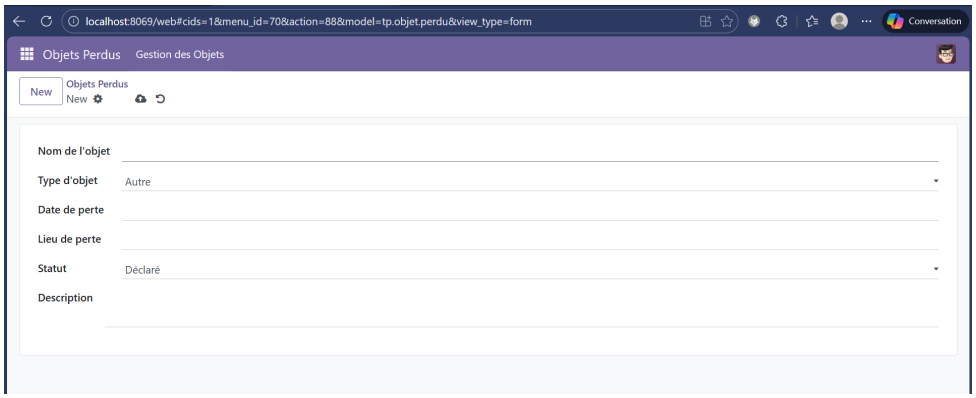

FIGURE 4.3 – Formulaire de déclaration d'un objet perdu

4.4 Affichage des objets enregistrés

L'affichage en liste permet d'avoir une vue d'ensemble rapide sur tous les objets stockés et leur état actuel de traitement au sein de l'organisation. Les utilisateurs peuvent utiliser des filtres dynamiques et des barres de recherche pour retrouver un objet spécifique à partir de son nom, de sa catégorie ou de sa date. Cette vue est essentielle pour la gestion quotidienne et permet de répondre instantanément aux demandes des propriétaires cherchant à récupérer leurs biens.

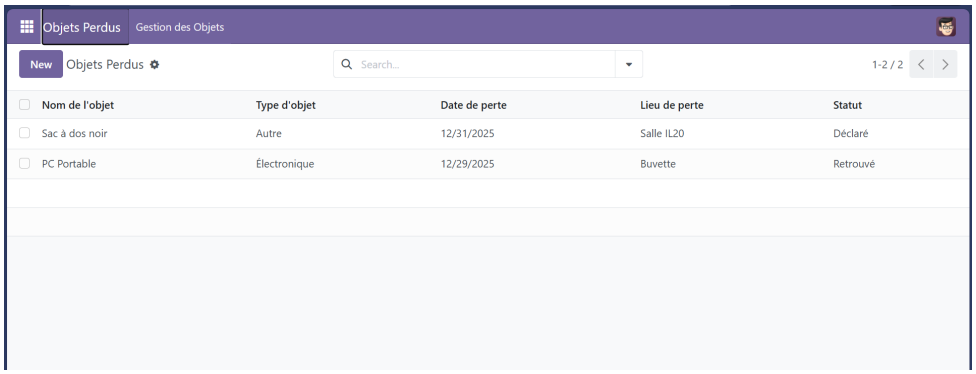

FIGURE 4.4 – Liste des objets perdus et retrouvés

4.5 Paramètres et configuration avancée

La configuration avancée permet d'ajuster les droits d'accès et d'activer le mode développeur pour inspecter les métadonnées techniques du système Odoo. C'est à travers cet écran que nous pouvons vérifier si les fichiers XML sont correctement chargés et si les séquences automatiques de numérotation fonctionnent sans erreur. Cette partie est réservée aux administrateurs système pour garantir la sécurité, la stabilité et la performance globale de l'application métier.

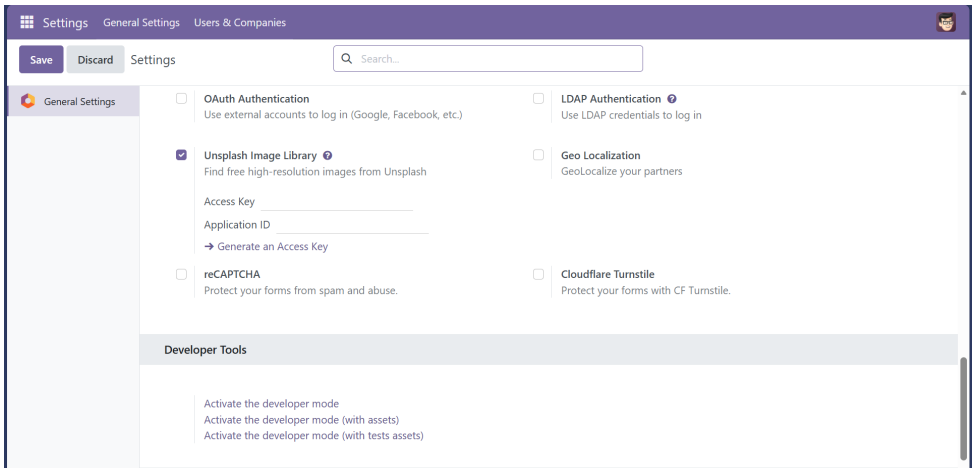

FIGURE 4.5 – Paramètres généraux et outils développeur

Chapitre 5

Conclusion Générale

La réalisation de ce module de gestion des objets perdus sous Odoo 17 a été une expérience technique et méthodologique particulièrement enrichissante à plusieurs niveaux. À travers ce projet, nous avons pu démontrer qu'une solution ERP moderne, lorsqu'elle est correctement personnalisée, peut transformer radicalement la gestion administrative d'un flux logistique. L'utilisation de Docker a apporté une rigueur supplémentaire dans la gestion de l'infrastructure, tandis que le framework Odoo nous a permis de nous concentrer sur la logique métier complexe sans nous soucier des couches techniques basses.

Ce travail nous a également permis de consolider nos compétences en programmation Python et en conception d'interfaces XML dynamiques au sein d'un environnement professionnel. Nous avons appris à modéliser des processus réels en objets informatiques cohérents, sécurisés et facilement exploitables par des utilisateurs finaux. Bien que le module soit pleinement fonctionnel, des perspectives d'évolution sont déjà envisageables, notamment l'intégration de notifications SMS automatiques. En conclusion, ce projet confirme que la digitalisation est un levier puissant pour l'efficacité organisationnelle moderne.